[Document title]

[Document subtitle]

[School]

[Course title]

Contents

[Introduction to the problem 2](#_Toc441749516)

[Analysis of the problem 2](#_Toc441749517)

[Planning algorithm 2](#_Toc441749518)

[Implementation design 2](#_Toc441749519)

[Testing cases and results 2](#_Toc441749520)

[Analysis of the results 2](#_Toc441749521)

[Instructions to execute the program 2](#_Toc441749522)

Linear planner with stack of goals

# Introduction to the problem

# Analysis of the problem

|  |  |  |  |
| --- | --- | --- | --- |
| Operator | Preconditions | Add | Delete |
| CleanOffice(o) | RobotLocation(o)  Dirty(o) Empty(o) | Clean(o) | Dirty(o) |
| Move(o1,o2) | RobotLocation(o1)  Adjacent(o1,o2) | RobotLocation(o2) | RobotLocation(o1) |
| Push(b,o1,o2) | RobotLocation(o1)  BoxLocation(b,o1)  Adjacent(o1,o2)  Empty(o2) | BoxLocation(b,o2)  RobotLocation(o2)  Empty(o1) | Empty(o2)  BoxLocation(b,o1)  RobotLocation(o1) |

Initial State

|  |  |  |
| --- | --- | --- |
| O1 | O2 | O3 |
| O4 Robot | O5 BoxA | O6 BoxB |
| O7 | O8 | O9 |

Final State

|  |  |  |
| --- | --- | --- |
| O1 | O2 | O3 |
| O4 | O5 | O6 BoxB |
| O7 | O8 BoxA | O9 Robot |

# Planning algorithm

The planning algorithm works as follows:

Current plan = empty  
Current state = initial state

We begin by putting in the stack of goals the final state and each of its subobjectives:

GoalStack.append(FinalState) // a single state  
ForEach E as a subobjective of the final state  
GoalStack.append(E) // add each predicate to the stack  
EndFor

The planner will analyse sequentially all the elements of the stack until it remains empty

We can found 4 different cases in the stack:

1. Operator
   1. Add the operator to the plan
   2. Apply the operator to the current state
2. Conditions list (preconditions of an operator)
   1. Check whether all are satisfied by the current state
   2. If any condition is not satisfied:
      1. Add the whole conditions list to the stack
      2. Add individually each condition not satisfied
3. Partially instanced condition (we don’t know the value of every single parameter)
   1. Search in the current state the value to be applied to the parameter in order to accomplish the condition
   2. Transmit the value found to all appearances of the parameter in the stack of goals.
4. Totally instanced condition
   1. Check if it can be verified in the current state
   2. If cannot be verified we should look for an operator that let us obtain the condition
   3. Add to the stack:
      1. first the operator
      2. then its list of preconditions
      3. each of its preconditions individually

# Implementation design

The operator is categorized with 3 lists:

1. The list of preconditions
2. The list of add
3. The list of delete

We will have 3 items important:

1. Current plan ( the plan in construction)
2. Current state
3. Stack of goals

The goal stack will contain a list of the different elements to be processed as a goal. It will contain 3 types of variables:

1. Operators
2. Individual subobjectives
3. Lists of subobjectives

Get verifying option procedure:

* Case CLEAN predicate:
  + Will be fully instantiated always
  + Return CleanOffice(o)
* Case BOXLOCATION predicate:
  + Can be achieved by a push

# Testing cases and results

# Analysis of the results

# Instructions to execute the program